

Appendix A

```

#include <analysis.h>
#include <ansi_c.h>

double g_dPressureCalc (double[], double[], double );

double g_dPressureCalc (double dIntensity[], double dCoefArray[], double dTemperature )
{
    int    count,
           i;

    double    * loc,
              * amp,
              * deriv,
              dPressure,
              TopHalf[1024];

    // Get Spectral Power density

    for (i=0; i<1024 ; i++) TopHalf[i] = dIntensity[1024+i];

    Spectrum (TopHalf, 1024);

    // Find the peak
    PeakDetector (TopHalf+8, 100, 50.0, 3, 0, 1, 1, &count, &loc,
                  &amp; , &deriv);

    if (count < 1 ) return (double) count;
    // Calculate the pressure
    //Polynomial Regression for CT20U15_C:
    //Y = A + B1*X + B2*X^2 + B3*X^3

    //Parameter Value Error
    //-----
    //A    17.62138    0.31925
    //B1   -0.50365    0.04074
    //B2    0.01343    0.00165
    //B3   -3.22463E-4  2.14632E-5
    //-----

    //    dPressure = 17.62138 - 0.50365 * loc[0] + 0.01343*loc[0]*loc[0] - 3.22463E-
    4*loc[0]*loc[0]*loc[0];
    dPressure = dCoefArray[0] +
                dCoefArray[1] * loc[0] +
                dCoefArray[2] * loc[0]*loc[0] +
                dCoefArray[3] * loc[0]*loc[0]*loc[0];

    // Apply temperature correction
    //Linear Regression for DATA11_B:
    //Y = A + B * X

```

```
//Parameter Value Error
//-----
//A   -1.39373    0.03399
//B   0.07041     7.93589E-4
//-----
dPressure = dPressure + dTemperature * dCoefArray[5] + dCoefArray[4];

return (dPressure);
}
```

Patent 37724.011000